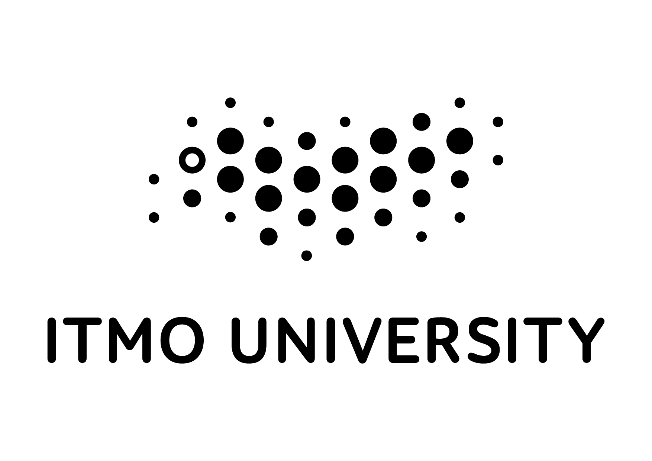
Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет систем управления и робототехники



Алгоритмы и структуры данных.

Отчет №4.

Выполнил студент гр. R32362

Лаптев Максим Сергеевич

Санкт-Петербург 2023

**Содержание**

1. Цель
2. Задача №1604 – В стране дураков
3. Задача №1726 – Кто ходит в гости
4. Задача №1521 – Военные учения
5. Вывод

**Цель**

Решить данные задачи, написав код, работа которого удовлетворяет условиям

**№1604 «В стране дураков»**

Изображение выглядит как диаграмма, текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

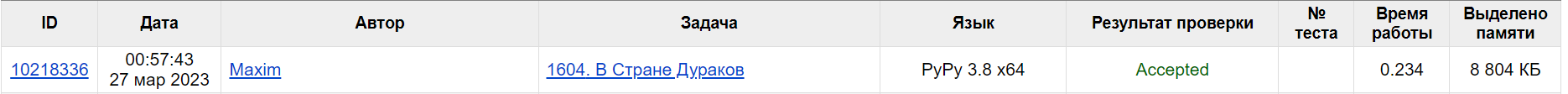
Исходный код:

k = int(input())  
x = [[str(index + 1), int(element)] for index, element in enumerate(input().split())]  
x = (sorted(x, key=lambda item: item[1], reverse=True))  
while len(x) > 1:  
 prev = 0  
 i = 1  
 while i < len(x):  
 if x[i][1] >= prev:  
 if x[0][1] >= x[i][1]:  
 print(x[0][0] + " " + x[i][0] + " ", end="")  
 x[i][1] -= 1  
 x[0][1] -= 1  
 prev = x[i][1]  
 if x[i][1] == 0: x.pop(i)  
 if x[0][1] == 0: x.pop(0)  
 else:  
 print(x[i][0] + " " + x[0][0] + " ", end="")  
 x[i][1] -= 1  
 x[0][1] -= 1  
 prev = x[i][1]  
 if x[i][1] == 0: x.pop(i)  
 if x[0][1] == 0: x.pop(0)  
 i += 1  
 x = (sorted(x, key=lambda item: item[1], reverse=True))  
if len(x) == 1:  
 print((x[0][0] + " ") \* x[0][1], end = "")

В задаче необходимо было как можно больше чередовать дорожные знаки. Для этого я отсортировал количества повторений, сохраняя при этом индексы в двумерном массиве. В цикле же я рассматривал пары значений с самого начала и если у них еще есть повторения, то выводил их. Те знаки, которых больше не осталось, удалялись из массива.

Язык программирования: python

Результат



**№1726** **«Кто ходит в гости»**

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Исходный код:

n = int(input())  
s = 0  
x, y = [], []  
for i in range(n):  
 x1, y1 = map(int, input().split())  
 x.append(x1)  
 y.append(y1)  
x.sort()  
y.sort()  
  
s = sum([(x[i+1]-x[i] + y[i+1]-y[i]) \* (i+1) \* (n-i-1) for i in range(n-1)])  
print( (s \* 2) // ((n-1)\*n))

Заметим, что ходить в гости можно только по дорогам. Это сильно упрощает задачу, ведь тогда доходить до одной точки можно обходя другие. А координаты х и у никак не связаны между собой. Реализуется это так: читаем все х и у, добавляем в массивы и сортируем. Теперь остаётся пробежаться по ним в цикле, считая разницу между следующими х и у и перемножая на (i+1)\*(n-(i+1)), так как именно столько раз пройдется конкретный путь. Теперь домножаем это на 2 и делим на количество путей – получаем среднюю длину пути.

Язык программирования: python

Результат



**№1521 «Военные учения 2»**

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Исходный код:

def remove(v, l, r, pos):  
 while l != r:  
 tree[v][0] -= 1  
 if pos <= tree[v \* 2][0]:  
 r = (l + r) // 2  
 v = v \* 2  
 else:  
 pos -= tree[v \* 2][0]  
 l = (l + r) // 2 + 1  
 v = v \* 2 + 1  
 tree[v][0] -= 1  
 return tree[v][1]  
  
def build(v, l, r):  
 if l == r:  
 tree[v] = [1, l + 1]  
 else:  
 mid = (l + r) // 2  
 build(v \* 2, l, mid)  
 build(v \* 2 + 1, mid + 1, r)  
 tree[v][0] = tree[v \* 2][0] + tree[v \* 2 + 1][0]  
n, k = map(int, input().split())  
tree = [[0] \* 2 for i in range(4 \* n)]  
build(1, 0, n - 1)  
current = k  
for i in range(n):  
 print(remove(1, 1, n, current), end=" ")  
 current = (current - 1 + k) % (n - 1 - i)  
 if current == 0: current = n - 1 - i

Для решения использовалась структура данных дерево отрезков. Для начала идёт его построение по информации о количестве солдат. Далее мы последовательно удаляем из него k солдат, в дереве оно ищется по алгоритму Binary Search Tree и удаляется. Тем временем переменная current меняет свое значение, идя «по кругу».

Язык программирования: python

Результат:



**Вывод**

В результате проделанной работы были успешно решены 3 задачи, используя язык программирования python . Работа программ выдает правильный ответ, а также удовлетворяет условиям задачи (проходит по времени и памяти).